CHARUSAT
CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

DEPSTAR

FACULTY OF TECHNOLOGY AND ENGINEERING

DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY AND RESEARCH

DEPARTMENT OF COMPUTER ENGINEERING

A.Y. 2023-24 [EVEN]

LAB MANUAL

CE266: SOFTWARE ENGINEERING

ID: 22DCE006-Probin Bhagchandani
22DCE011-Nisarg Chaudhari
22DCE018-Urva Dave
22DCE040-Japan Kachhiya

PRACTICAL-8

AIM: Practicing with 'Version Control System-GIT'. Create repository and mention the demos of push command and Pull request.

Reference link: https://opensource.com/article/19/5/practical-learning-exercise-git Explore the Jenkins DevOps tool as a continuous Integration and continuous deployment tool. https://www.guru99.com/jenkins-tutorial.html

1. Practicing with 'Version Control System-GIT'. Create repository and mention the demos of push command and Pull request.

Creating a Git Repository:

1. Install Git:

If you haven't already, download and install Git from the official website.

2. Open Terminal (Linux/Mac) or Command Prompt (Windows):

Navigate to the directory where you want to create your Git repository.

3. Initialize a New Repository:

Run the following command to initialize a new Git repository:

git init my-project

cd my-project

4. Add Files and Commit Changes:

Create some files in your project directory, or copy existing files into it.

touch README.md

Add these files to the staging area and commit them:

git add .

git commit -m "Initial commit"

- Push Command:

1.Create a Remote Repository:

- Go to a Git hosting service like GitHub, GitLab, or Bitbucket.
- Create a new repository (e.g., "my-project") on the hosting platform.

2.Link Remote Repository:

- Link your local repository to the remote repository you just created.

git remote add origin <remote_repository_url>

3.Push Changes:

git push -u origin master

- Pull Request:

1. Make Changes:

- Make some changes to your project locally.

2.Commit Changes:

- Stage and commit your changes:

git add .

git commit -m "Made some changes"

3.Push Changes:

- Push your changes to your remote repository:

git push origin master

4. Create Pull Request:

- Visit your remote repository on the hosting platform.
- Click on the "Pull Request" or "New Merge Request" button.
- Compare the changes between your branch and the master branch.
- Create the pull request, adding a description if necessary.

2.Explore the Jenkins DevOps tool as a continuous Integration and continuous deployment tool.

Jenkins is a widely used open-source automation server that enables continuous integration (CI) and continuous deployment (CD) in software development. It helps automate the process of building, testing, and deploying software, allowing teams to deliver high-quality code more efficiently. Here's an overview of how Jenkins facilitates CI/CD:

1. Continuous Integration (CI):

Automated Builds:

Jenkins can automatically trigger builds whenever changes are pushed to the version control system (e.g., Git). It fetches the latest code from the repository and initiates the build process.

2. Build Pipelines:

You can create complex build pipelines in Jenkins, defining sequential stages for building, testing, and packaging the application. Each stage can execute specific tasks, such as compiling code, running unit tests, and generating artifacts.

3. Testing Integration:

Jenkins integrates with various testing frameworks, allowing you to run automated tests as part of the CI process. This ensures that code changes are thoroughly tested before being merged into the main branch.

4. Static Code Analysis:

Jenkins supports plugins for static code analysis tools like SonarQube and Checkstyle. These tools can analyze code quality, identify potential issues, and enforce coding standards.

5. Feedback Loop:

Jenkins provides real-time feedback on build status, test results, and code quality metrics. Developers can quickly identify and address any issues, leading to faster iteration cycles.

6. Continuous Deployment (CD):

Deployment Pipelines:

Jenkins facilitates the automation of deployment pipelines, enabling seamless deployment of applications to various environments (e.g., development, staging, production). You can define multiple stages in the pipeline, each responsible for deploying to a specific environment.

7. Rolling Deployments:

Jenkins supports rolling deployments, allowing you to deploy new versions of the application gradually while monitoring for any issues. This helps minimize downtime and mitigate risks associated with deploying changes.

8. Integration with Configuration Management Tools:

Jenkins integrates with configuration management tools like Ansible, Chef, and Puppet, enabling automated provisioning and configuration of infrastructure as part of the deployment process.

9. Continuous Monitoring:

Jenkins can trigger automated tests and health checks after deployment to ensure that the application is functioning as expected. It can also integrate with monitoring tools to monitor application performance and detect any anomalies.

Continuous Feedback:

Jenkins provides visibility into the deployment process, including deployment logs, metrics, and alerts. This feedback loop enables teams to continuously improve the deployment process and address any issues proactively.

Jenkins is a powerful tool for implementing CI/CD practices, enabling teams to automate the entire software delivery lifecycle. By integrating with version control systems, testing frameworks, and deployment tools, Jenkins helps teams deliver high-quality software faster and more efficiently, ultimately improving collaboration, productivity, and customer satisfaction.

CONCLUSION: By delving into version control with Git and exploring Jenkins as a tool forcontinuous integration and deployment, the aim is to bolster proficiency in modern software development practices, fostering efficient collaboration and deployment workflows**.**

Staff Signature:

Grade:

Remarks by the Staff: